MASSACHUSETTS INSTITUTE OF TECHNOLOGY
A. I. LABORATORY

Artificial Intelligence
Memo No. 241

February 1971

AN A.I. APPROACH TO ENGLISH MORPHEMIC ANALYSIS

Terry Winograd*

ABSTRACT

This paper illustrates an approach toward understanding natural language through the techniques of artificial intelligence. It explores the structure of English word-endings both morpho-graphemically and semantically. It illustrates the use of procedures and semantic representations in relating the broad range of knowledge a language user brings to bear on understanding and utterance.

Work reported herein was conducted at the Artificial Intelligence Laboratory, a Massachusetts Institute of Technology research program supported in part by the Advanced Research Projects Agency of the Department of Defense and monitored by the Office of Naval Research under Contract Number N00014-70-A-0362-0002.

*Assistant Professor of Electrical Engineering, M.I.T.

Section 1. Introduction

This paper presents an analysis of some aspects of English morphemic structure. It is based on an "artificial intelligence" approach to language, and differs from many linguistic descriptions in its basic premises of how language can be described. Stated over-simply, they are:

1. The structure of language can best be described by modelling the interpretation rather than the generation of utterances.
2. Many parts of the model are best characterized as algorithms or procedures which explicitly carry out the interpretation process, rather than process-free rules which implicitly control an interpretive or generative process.
3. It is impossible to describe major areas of language behavior with one level of analysis, such as syntax, isolated from other levels such as semantics.
4. A linguistic theory should describe how a speaker integrates all of his different areas of knowledge in language use. This should include syntax, phonology, semantics, and their relation to the speaker's knowledge of the context of the utterance and the subject of discourse.

The framework for expressing the linguistic analysis is a computer program for understanding natural language. [Winograd 1971]. It is an integrated system which combines syntax, semantics, and deduction within a framework of special languages designed for expressing linguistic regularities. It has the services of a powerful general-purpose deductive language called PLANNER [Hewitt 1969], which can be used for reasoning both about the linguistic forms and the subject matter

being discussed.

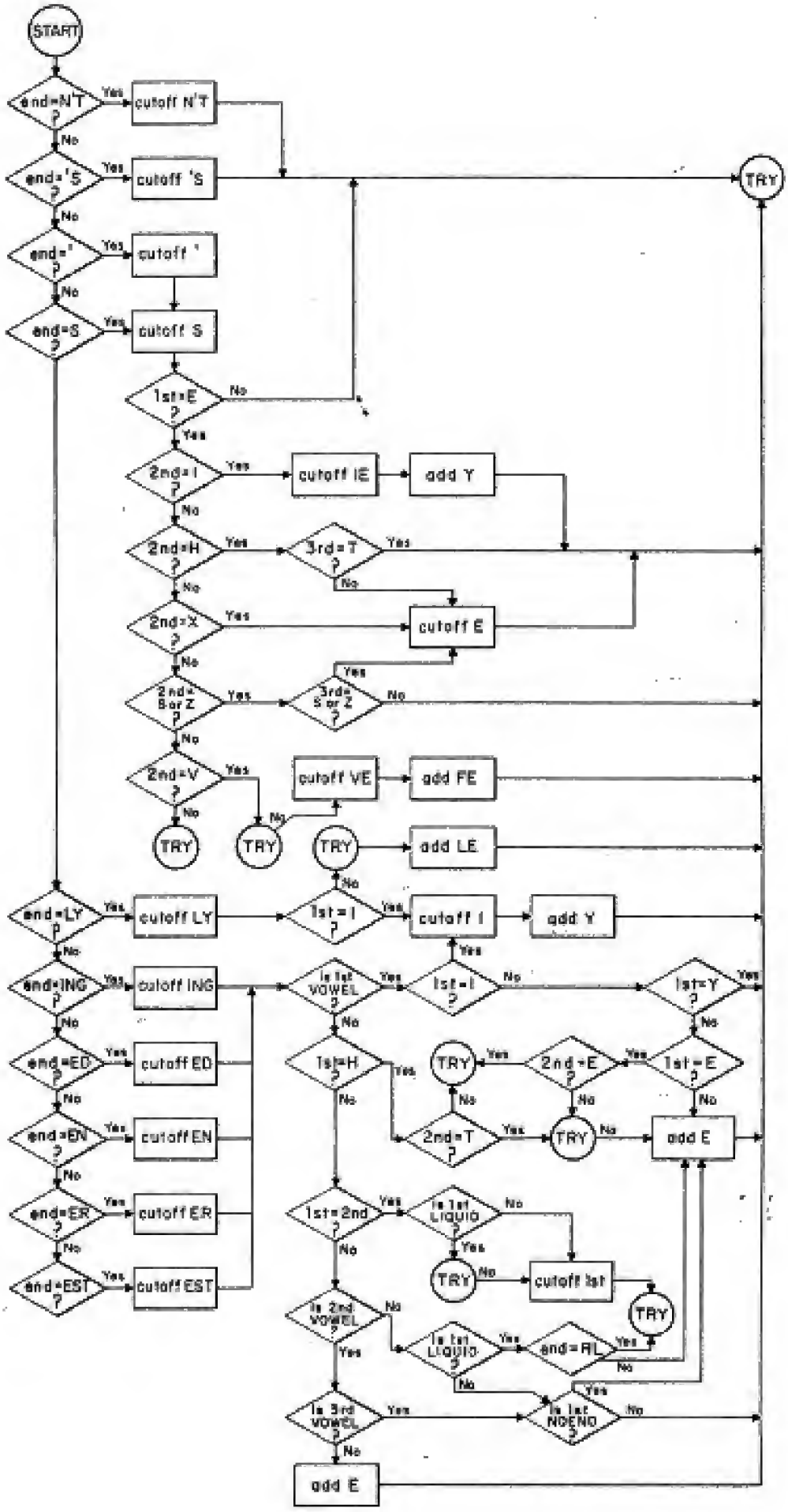
The analysis of morpho-graphemic structure given in Section 2 is an expanded version of the one already used in the system. Section 3 describes an analysis of some semantic aspects of English word formation, and shows how it would relate to the rest of the system. It is important to recognize that this is not simply a proposal for a particular computer program. Many of the ideas have grown from the use of the computer, but the basic concept of procedural descriptions is independent of any actual computer implementation. It is a flexible and useful formal representation for many phenomena, and could well be used in linguistic descriptions and theories which had no connection to the direct use of computers.

Section 2. The "Spelling" of English Word Endings

English has a complex set of "spelling rules" which describe the way inflectional endings are attached to root words. These rules enable a reader to recognize that, for example, "pleasing" is a form of "please", while "beating" is a form of "beat". There is a structure of conventions for doubling consonants, dropping "e", changing "y" to "i", etc. when adding endings. For spoken language these would be called the morpho-phonemic rules, while for written language, we can call them morpho-graphemic.

In any complete analysis of language, a word like "running" should not be considered a completely separate lexical entity. It is a regular inflected form of "run", and both its form and usage can be derived from "run" and an analysis of the inflection. First, the analysis should include a formalism for describing the way the actual sequence of letters in the inflected form is related by regular rules to the form of the root word. There are various ways this might be expressed, describing the generation or analysis of the resultant forms. One formalism which seems particularly well suited is an algorithm for interpreting inflected words. The regularities are expressed directly within a process which accepts an unfamiliar word as an input, and "breaks it down" into a familiar root and an ending.

The flow chart in Figure 1 describes a procedure designed to handle a number of regular endings: "-n't" for negative; "-'s" and "-'" for possessive; "-s" and its various forms for plural nouns and singular third-person verbs; "-ing", "-ed", and "-en" verb


START
end=N'T ?
Yes
cutoff N'T
No
end='S ?
Yes
cutoff 'S
No
end=' ?
Yes
cutoff '
No
end=S ?
Yes
cutoff S
1st=E ?
No
Yes
2nd=I ?
Yes
cutoff IE
add Y
No
2nd=H ?
Yes
3rd=T ?
Yes
No
2nd=X ?
Yes
cutoff E
No
2nd=S or Z ?
Yes
3rd=S or Z ?
Yes
No
No
2nd=V ?
Yes
cutoff VE
add FE
No
TRY
TRY
TRY
add LE
TRY
end=LY ?
Yes
cutoff LY
1st=I ?
Yes
cutoff I
add Y
No
Yes
No
end=ING ?
Yes
cutoff ING
Is 1st VOWEL ?
Yes
1st=I ?
No
1st=Y ?
Yes
No
No
end=ED ?
Yes
cutoff ED
1st=H ?
Yes
TRY
Yes
2nd=E ?
Yes
1st=E ?
No
No
No
No
end=EN ?
Yes
cutoff EN
2nd=T ?
Yes
TRY
No
add E
No
end=ER ?
Yes
cutoff ER
1st=2nd ?
Yes
Is 1st Liquid ?
No
No
end=EST ?
Yes
cutoff EST
Yes
TRY
No
cutoff 1st
TRY
Is 2nd VOWEL ?
No
Yes
Is 1st Liquid ?
Yes
end=RL ?
Yes
No
No
Yes
Is 3rd VOWEL ?
Yes
Is 1st NOENO ?
No
No
add E

forms; the superlative "-est"; the comparative "-er"; and the adverbial "-ly".

The description uses a few simple notations in addition to normal flowchart conventions. "No" and "yes" answers to branching conditions are represented by single and double arrows respectively. The function CUTOFF indicates what letters are to be removed from the end of the word. The ordinals "1st", "2nd", etc. count letters backwards from the end of the word, including only those which have not been cut off. Several subclasses of letters are used -- VOWEL is (A E I O U Y), LIQUID is (L R S V Z), and NOEND is (C G S V Z). The label TRY represents the part of the procedure which tries looking up the supposed root in the dictionary. It uses whatever letters have not been cut off. At some places, the program tries an interpretation, then if that fails, carries out a different analysis and tries again.

As the flowchart shows, these endings share many aspects of morpho-graphemic structure, and the procedural representation is able to capture these generalities by having them share parts of the procedure. It can also detail those aspects peculiar to each ending. This is not a complete description of how these endings appear, but covers the great majority of words, and could easily be expanded to treat more special cases. The program can correctly analyze such relationships as: bashes - bash, bathes - bathe, leaning - lean, leaving - leave, dented - dent, danced - dance, dogs - dog, kisses - kiss, curved - curve, curled - curl, rotting - rot, rolling - roll, played - play, plied - ply, realest - real, palest - pale, knives - knife, prettily - pretty, nobly - noble.

It is important to note the way in which exceptions are handled in the approach. Since the approach is interpretive, this procedure

can be viewed as one piece of a more general procedure which accepts an English utterance and tries to understand it. Morpho-graphemic analysis takes place whenever the procedure encounters a word token which it cannot find in the lexicon. An irregular form like "was" is in the lexicon directly, and the rules will never be applied in trying to either analyze or produce it. This distinction between lexical idiosyncracies and morpho-graphemic generalities is empirical. A productive rule like the one relating "sing" and "sang", "ring" and "rang", might well be included in a more complete procedural description, while less productive ones (such as relating "will" to "won't") will be left in the form of separate lexical entries.

It is tempting to see this formalism as a simple finite state machine, but this is not applicable for several reasons. First, the tests which can be done to a word in deciding on its form are not, in general, simple checks of the next input letter. Whether a certain analysis is possible may depend, for example, on how many syllables there are in the word, or on some complex phonological calculation involving vowel shifts. Semantic and syntactic information are also applicable as will be described later. In the case of multiple endings (as in "patronizingly") the program must be applied recursively, and must take into account the entire derivation up to the point of analysis. The simplified version shown in Figure 1 does not account for these extra levels of complexity, but the view of the procedure as a flowchart for a part of an integrated computational system makes it easy to integrate them.

The straightforward morpho-graphemic analysis alone is sufficient to do much of the interpretation. In fact some systems (e.g. [Thorne 1969]) use it to avoid having a dictionary of open class words. The inflection

of a word in the input determines its possible syntactic classes.

People rarely operate at this level of ignorance. They use their lexical knowledge to recognize that "under" is not a comparative form of some adjective "und", and that "bely" is not an adverbial form of "be". This type of knowledge can readily be integrated into the interpretive program. Once a possible analysis is determined, the hypothetical root can be checked in the dictionary. If it is not there, the program can try another analysis. Notice in Figure 1 that a word ending in a double LIQUID, followed by an inflection, is first tried with the doubled consonant (as in "rolling - roll"), then if that is not found, another try is made without the doubled consonant (as in "patrolling - patrol").

In one sense, this "multiple guess" method represents an open place in the morpho-graphemic theory. No prediction is made about which form will succeed. A more sophisticated test might be included which looked at the number of syllables in the word, deciding that a single-syllable word, such as "roll" has the consonant doubled, while the multi-syllabic "patrol" does not. It could go further and recognize that "unroll" is a derivative of a one-syllable word, and therefore behaves like "roll". Within the procedural formalism, such knowledge can be included by adding to the procedure calls to further phonological, syntactic, or semantic subroutines.

Whatever the level of analysis in such a theory, the remaining details are included as idiosyncracies of the individual lexical items involved. People in general operate at a much lower level of analysis than linguists, relying on lexical idiosyncracies to provide information which might

instead be represented as generalities involving diachronic linguistic facts (e.g. knowing the Old English or Latin origin of a word, and understanding the effects of this origin on its morphology). A theory should be able to allow this vague borderline if it is to describe the actual competence of a speaker.

One simple relevant bit of knowledge about a word is its syntactic class. If a possible root is found in the dictionary, its class can be checked to see if the ending is appropriate (e.g. an "-est" can apply only to an adjective). Of course, a word can be in more than one category, and all of the possibilities must be considered in the analysis. This check of syntactic class can be included directly into the intepretation procedure, just as the other types of knowledge described above.

Section 3. Some Semantic Aspects of Nominalization

The description above includes only a few of the possible endings in English. In particular, it involves only those which involve no major semantic change to the root. There is a whole world of prefixes and suffixes which modify words, changing both their syntactic class and their meaning. A "baker" is a person who bakes, a "painting" is an object which is the result of painting, or may be an event ("the painting of the house went without trouble."). An "invention" is a thing which is invented, "detainment" is an event of being detained, and so forth.

The procedure described in Figure 1 could easily be augmented to include the morphographemic behavior of endings like "-tion" and "-ment". It already includes those such as "-er" and "-ing". What must be added is some formal description of the semantic changes and relationships implied by the additional morpheme.

One of the central committments of the theory of language described here is the existence of a formalism for representing the "meaning" of words and utterances with respect to an internal "model of the world" held by the language user. This internal model contains a set of interrelated "concepts" and procedures for manipulating these in carrying out deductions. Our description of language uses a logical-deductive language (PLANNER) to represent this model. This paper will not describe the formalism or its justification in detail, but will use parts of it to illustrate how problems such as nominalization might be handled. (See [Winograd 1971]for much more detail).

The dictionary definition of a word may include several meanings. Each of these involves a basic expression in the logical formalism, and a set of semantic markers (see [Katz and Fodor 1964]) as preliminary filters for proposed semantic combinations. The exact status of these markers will be discussed later.

One possible meaning for bake would involve "cooking in an oven", and would be done by a person to some sort of food. The resulting formalism might be (in a simplified version of the notation used in [Winograd 1971]):

(VERB ((*PERSON) (*FOOD)) (COOK #1 #2 OVEN))

The character "*" is used to prefix a semantic marker, and the symbols "#1" and "#2" are used to represent the subject and object of the verb. Within the total language understanding system, this definition is applied after the actual semantic (deep structure) subject and object have been determined. The semantic markers check its applicability to the particular objects involved, and the resulting logical expression (with these objects substituted for "#1" and "#2") is used in building up the "semantic structure" which representes the meaning of the sentence.

There is no necessity to express "bake" in this way. The model might include "baking" as a primitive concept, rather than as an instance of "cooking", and relate the two within the deductive knowledge of the world-model. There is no fixed line between the knowledge which represents the "definition" of a word, and the knowledge which relates that definition to the rest of the language-user's world.

Returning to the problem of word-endings, we could define an interpretive procedure associated with the "-er" ending, which operates on the definition of a verb to produce a definition for the corresponding noun. The "-er" program might say something like "If the verb has an interpretation which involves a person as its subject, then the noun represents a person who does that action." This would be described within a formal semantic language which has primitives specially suited

for manipulating these semantic definitions. The exact form of these primitives is a part of a complete theory of semantics, just as a particular set of primitive "transformations" is inherent to a transformational theory of syntax.

The result of analyzing "baker" would then be a noun definition (again in simplified notation):

```
(NOUN    (*PERSON)  ((COOK  ***  ?  OVEN)) )
```

Here, the symbol "***" represents the object being described by the noun, and the "?" indicates an unspecified object. The definition is of an object with the semantic marker "*PERSON", and which cooks some object in an oven. A more detailed description (see [Winograd 1971] Chapter 4) would show how this new definition might include additional information, such as retaining the semantic marker restriction of *FOOD on the object being baked, or putting a specific time into the abstract relationship of "cook". The details for doing this already exist in the language-understanding system, and could be included in the simple program which performs the semantic analysis of the "-er" ending. The program of course operates on other verbs in the same way, for example converting the definition of "run":

```
(VERB   ((*PERSON))   (RUN #1)   )
```

to "runner":

```
(NOUN    (*PERSON)  ((RUN ***))   )
```

A similar program for "-ing" might convert "shoot":

```
(VERB   ((*PERSON) (*OBJECT))   (SHOOT #1 #2)  )
```

into "shooting":

```
(NOUN   (*EVENT)   ((SHOOT  ?   ?))   )
```

(again, there are other details of how the event would be referred to in the logical representation, etc.). Note how this might fit in with the use of a phrase beginning with "of". One definition of "of" would say (using the semantic primitives) "If the "of" phrase modifies an event with unspecified objects, the object of the preposition should be substituted for one of the objects." Inherent in this definition is the ambiguity of a phrase like "the shooting of the hunters", since there are two unspecified objects in the definition of "the shooting", and the "of" phrase might apply to either.

The definition of "-er" given above is not very sophisticated. You may well say you are not a "baker" even though you have baked things. The nominalized form includes the additional idea of doing the activity as an occupation. This subtlety can be included by slightly modifying the "-er" program, and using the ability of the logical formalism to create an arbitrary name for a relationship or event, and refer to that name in other relationships. If these arbitrary names are of the form "REL" followed by a number, the resulting definition of "baker" might be:

(NOUN (*PERSON) ((COOK *** ? OVEN (REL65))(OCCUPATION *** REL65)))

The name REL65 is assigned to the relationship involving the person's cooking something in the oven, and the second relation shows that the first indeed represents the person's occupation.

The program now defined for "-er" is more complete, but it ignores the difference between a "baker" and a "drinker". "Drink" is also a verb with a human subject, but a "drinker" is not a person who drinks for a living. In this case, the "-er" ending represents habitual action.

It is tempting to once again make use of semantic markers, and classify the verbs "bake" and "drink" into two different categories. The "-er" program could then decide which definition was applicable by having an appropriate semantic marker restriction for each. This is a dangerous route. For each new type of nominalization, we may need to introduce specialized classifications of verbs, whose only purpose is to determine the meaning of that particular nominalization.

One way to understand the limitations of a system of semantic markers is to see them as a type of "pre-calculation". In a semantic system, a definition may include a proviso like "This meaning of the verb applies only if the subject is a *PERSON". This could be implemented by having a deductive procedure which is called each time the definition is used, and which uses facts about the linguistic forms and about the world being discussed. The convenience of semantic markers comes in noticing that most words can be associated "permanently" with an outcome of this deduction. The word "boy" will refer to an object which can be assumed to be a person simply because of the word used, and without further deduction. In addition, many of the required computations can be reprsented in a subclass-inclusion tree, so their outcomes can be related by simple logical relations. If the *PERSON deduction is true for an object, then the *ANIMATE deduction must be true, and the *PHYSICAL-OBJECT deduction, etc.

Setting up a tree of semantic markers is then pre-calculating a related set of predicates concerning the objects referred to by various words. This works quite well as a computational convenience for simple objects and calculations. However it fails in two ways for more complex and realistic ones. First, as mentioned above, the actual test

needed to decide whether a given definition is applicable may be quite complex, and limited in use. It seems unsatisfactory to assume that predicates like "activity which can be done for a living" have been pre-computed and stored in a gigantic marker-tree for each word in the lexicon.

More important, the basic idea of semantic markers depends on the possibility of computing the value of the predicate on the basis of the words, outside of any particular context. One of the most important aspects of language use is the fact that every utterance occurs in a context of a situation and other utterances. The speaker or writer makes heavy use of this in the way he communicates. The decision as to what activities can be done for a living may depend on the situation. In a discussion about people who test products for food companies, a "drinker" might well have a good job. If an artist is composing a picture of a group of people, he might well "raise one of the children". In this context, the usual meaning of "raising children" is inapplicable, because the objects being referred to with the word "children" are in fact drawings.

This is often explained by assuming that a word has a "primary" meaning, and other uses are not explained by the semantic theory, but are done by "analogy" or "extension" of the basic rules. By recognizing the computational nature of semantic markers, there is no need to make this artificial distinction. The selection restrictions on syntactic combinations depend not on the specific English words, but on the semantic features of the objects and relationships actually being described. Pre-calculated semantic markers are a convenient device for handling simple cases, but are only a first-order approximation to the semantic organization of language.

The actual program for interpreting an ending like "-er" will include not only ways of manipulating the dictionary definitions described above, but also calls to a deductive program which can examine the world-model. It can use the context and knowledge of an "intelligent" reader to answer the semantic questions needed to interpret a phrase or complex word. Rather than asking "Does the noun have the semantic marker *CHILD", it can ask "Is the object being referred to by the noun actually a child?"

Section 3. Some Theoretical Comparisons

The analysis presented above differs in a number of ways from more traditional linguistic descriptions. It is important to note which aspects represent actual departures in theory, instead of notational changes.

First, the idea of an interpretive rather than the usual generative grammar does not really represent a theoretical disagreement. A "generative grammar" in the sense originally intended is "a system of rules that in some explicit and well defined way assigns structural descriptions to sentences" [Chomsky 1965, p.8]. A program for interpretation serves as such an explicit and well-defined set of rules in the same way as a set of productions and transformations. This interchangeability is particularly easy to see in the case of a finite-state grammar, which can be equally well defined by a set of productions or by the description of a finite-state acceptor. It applies as well to more complex languages.

The advantage of viewing language through interpretation is not one of theoretical power, but of convenience and simplicity of description. It is particularly imporant in interconnecting the various aspects of language - syntax, semantics, phonology, etc. in a meaningful way.

In some senses, the programs described above act very much like "transformations". The "-er" procedure transforms a semantic definition into "bake" into one for "baker" following a fixed algorithm. There are two basic differences between this and the usual sense of "transformation" in linguistics. First, as described in the previous section, the algorithm may include computations which involve all levels of analysis, including the phonological form of the word, or the semantic features of the object or objects it refers to.

This does not mean that the rules are any less "explicit" or "well defined", but that they have a much wider scope. The belief underlying this is that no set of rules operating at one level can adequately describe language. The interaction between levels is critical to the description of each of them.

Second, the transformations operate on an explicit semantic rather than syntactic representation of meaning. There is no attempt to have some syntactic "deep structure" represent all of the meaning. For example, in the case of "baker", there must be some explanation for the addition of the meaning "as an occupation". "Baker" cannot be derived by a syntactic transformation from "a person bakes", since syntactic transformations cannot add meaning. It therefore must come from a deep structure which involves a sentence like "a person does S as an occupation". Similarly, each semantic fact about the interpretation of an ending must be expressed by making up deep structures with hypothetical additional sentences to express the meaning. See [Lees 1960] or [Chomsky 1969] for a description of many of the difficulties involved in describing English nominalization.

The main thesis of this "artificial intelligence" approach to describing language is that it is both unnecessary and unwise to try to separate the syntax of language from its meaning, or its meaning from context and non-linguistic knowledge. The importance of language lies in its ability to interrelate these, and its structure is organized to achieve that purpose. By using tools such as procedural descriptions, a linguist can deal explicitly with the interconnections, and describe language as an organized system for the communication of meaning.

References

1. [Chomsky 1965] Chomsky, Noam, ASPECTS OF THE THEORY OF SYNTAX, M.I.T. Press, 1965.

2. [Chomsky 1969] Chomsky, Noam, "Remarks on Nominalization", in Jacobs and Rosenbaum (eds.) READINGS IN ENGLISH TRANSFORMATIONAL GRAMMAR.

3. [Katz & Fodor 1964] Katz, J. J., and Fodor, J.A., "The structure of a Semantic Theory," in Fodor and Katz (eds.) THE STRUCTURE OF LANGUAGE.

4. [Hewitt 1969] Hewitt, Carl, "PLANNER: A Language for Proving Theorems in Robots," Proceedings IJCAI, 1969, pp. 295-301.

5. [Lees 1960] Lees, Robt., THE GRAMMAR OF ENGLISH NOMINALIZATIONS, Bloomington Indiana, 1960.

6. [Thorne 1969] Thorne, J., "A Program for the Syntactic Analysis of English Sentences," CACM 12:8 (August 1969), pp. 476-480.

7. [Winograd 1971] Winograd, Terry, "Procedures as a Representation for Data in a Computer Program for Understanding Natural Language" AI-TR1 Artificial Intelligence Laboratory, M.I.T.